Ada Language Server unbonded memory growth.

ADACORE SECURITY ADVISORY

Document Id: SEC.ALS-1818

Version 1 - Jun 30, 2026

AdaCore

Title	Ada Language Server unbonded memory growth.	
Status	Final	
Author	Thomas Serabian	
Reviewed by	Frédéric Léger	

Revision History

Version	Date	Comments
1	Jun 29, 2026	Initial description

Contents

1. Preface

1.1. Scope

This document is an advisory describing the security impact of `ALS-1818`. The issue is tracked under the ticket number `ALS-1818` in AdaCore's issue tracking database.

This document also presents possible workarounds and mitigation for the issue.

1.2. Distribution

This advisory is made available in confidence to AdaCore customers under embargo until 2026-08-15 so that they can address the issue it describes before public availability.

Thereafter, it will be available to the general public under the terms of the `CC BY-ND 4.0` license.

1.3. Contact

For questions on this document, please contact AdaCore support at product-security@adacore.com or using the standard reporting procedures if you are an AdaCore customer.

2. Vulnerability

2.1. Affected Products

The vulnerability described in this document was reported for the following product versions:

- *Ada Language Server* - `ada_language_server` *24.0* (20230829)

The vulnerability described in this document applies to the following product versions:

- `ada_language_server` <= 26.2

2.2. Severity and Impact

CVSS v3.1 score: 7.5 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H)

This vulnerability allows a *Remote Deny of Service* attack on an *Ada Language Server* if an attacker sends duplicate or invalid request IDs that crash the server.

A CVE (*Common Vulnerabilities and Exposures*) has been created for that case, and will be referred as `CAN-2026-2031774` (until MITRE delivers a real CVE ID).

2.3. Detailed Description

Thanks to the investigation work done by Jon Hood, a vulnerability on *Ada Language Server* has been discovered.

Due to a regression introduced in commit e71b3dbe (2023-08-29), `LSP.Known_Requests.Remove_Request`` fails to dispatch its message. As a result, the visitor branch (On_Server_Request) that would normally delete the completed request from the map is never executed. This creates a multi-layered vulnerability profile:

- Unbounded Memory Growth (CWE-401, CWE-770): The in-flight request map grows monotonically for the lifetime of the server.
- Use-After-Free (CWE-416): Every map entry consists of an Unchecked_Access cast of a Server_Request that is freed shortly after insertion. A subsequent `$/cancelRequest` whose, id matches a completed request results in a UAF write (Element `(Cursor).Canceled := True`) into freed heap memory.
- Denial of Service (CWE-617): If any LSP client reuses a request id after the prior request has completed, it triggers a `CONSTRAINT_ERROR` from `Hashed_Maps.Insert`. This exception is swallowed by the `Input_Task`'s blanket handler, which tears down the server entirely.

This bug is reachable by any entity capable of speaking the Language Server Protocol to the process (e.g., local editor processes, extensions, or network peers if exposed over a socket).

3. Solution

3.1. Workarounds

No workaround is available

3.2. Correction

The vulnerability described in this document is corrected in the following product versions:

- *Ada Language Server* - `ada_language_server` >= 26.2

Starting from build date `20260604`, the *Ada Language Server* component is fixed on `wavefront`.

4. Appendix

4.1. CVSS Score Justification

Metric	Justification
AV:N	The vulnerable component is bound to the network stack and the set of possible attackers extends beyond the other options listed in *Attack Vector: Adjacent*, up to and including the entire Internet.
AC:L	The attacker only needs to send duplicate or invalid requests ID which will trigger the failure
PR:N	There is no privilege needed.
UI:N	The vulnerability can be exploited purely programmatically without requiring any action from a human user.
S:U	The impact is entirely contained within the LSP server and does not affect the underlying operating system.
C:N	The flaw does not provide a mechanism to read or exfiltrate private data or memory contents.
I:N	The bug causes memory corruption but cannot be leveraged to modify system data or inject unauthorized state.
A:H	Exploitation results in a total denial of service via unbounded memory growth or an immediate application crash